

Accelerating Data Regeneration for Distributed Storage Systems with Heterogeneous Link Capacities

Yan Wang, Xunrui Yin, Dongsheng Wei, Xin Wang, *Member, IEEE*, Yucheng He, *Member, IEEE*

Abstract—Distributed storage systems provide large-scale reliable data storage services by spreading redundancy across a large group of storage nodes. In such a large system, node failures take place on a regular basis. When a storage node breaks down, a replacement node is expected to regenerate the redundant data as soon as possible in order to maintain the same level of redundancy. Previous results have been mainly focused on the minimization of network traffic in regeneration. However, in practical networks, where link capacities vary in a wide range, minimizing network traffic does not always yield the minimum regeneration time. In this paper, we investigate two approaches to the problem of minimizing regeneration time in networks with heterogeneous link capacities. The first approach is to download different amounts of repair data from the helping nodes according to the link capacities. The second approach generalizes the conventional star-structured regeneration topology to tree-structured topologies so that we can utilize the links between helping nodes with bypassing low-capacity links. Simulation results show that the flexible tree-structured regeneration scheme that combines the advantages of both approaches can achieve a substantial reduction in the regeneration time.

Index Terms—Regenerating codes, heterogeneity, distributed storage systems, erasure codes, fault tolerance.

I. INTRODUCTION

Large-scale distributed storage systems are widely used today to provide reliable data storage services, by spreading data redundancy over a large number of storage nodes. Users can access their data anytime and anywhere. Such application scenarios include large data centers such as Google File Systems, Total Recall, OceanStore and peer-to-peer storage systems Wuala.

In distributed storage systems, the level of redundancy is usually described by parameters (n, k) , where n is the number of storage nodes holding coded blocks of a file, and k indicates that the file can be reconstructed from any k out of the n storage nodes. For example, we can use an (n, k) maximum distance separable (MDS) code to encode a file of size M into n blocks of equal size M/k , and disseminate them to n storage nodes with each holding one block. Then, the original file

can be reconstructed from any k storage nodes. In literature, the ability to reconstruct the file from any k storage nodes is usually referred to as the *MDS property* [1].

In such big systems, nodes fail frequently, and the failures should be handled on a routine basis. After a node fails or leaves the system, the reliability degrades, and the protected data becomes vulnerable. To maintain the same level of redundancy, it is important to regenerate the lost data at a replacement node, called *newcomer*, as soon as possible [2]. In this work, we focus our attention on how to minimize regeneration time in the regeneration process.

An intuitive way to achieve a minimized regeneration time is to minimize the total amount of data transferred for regeneration, which is called *repair bandwidth*. In this direction, Dimakis *et al.* proposed Regenerating Codes to achieve the minimum repair bandwidth [3]. For simplicity, they assumed that each surviving node participating in the regeneration, called *provider*, sends the same amount of repair data to the newcomer. In this paper, we focus on the regenerating codes with functional repair, where the regenerated data may be different from the lost data. We will use the term *repair traffic* to denote the amount of repair data transmitted from each provider to the newcomer.

Under the assumption of uniform repair traffic, Demakis *et al.* derived the minimum repair bandwidth to maintain the MDS property. They found that there is a trade-off between repair bandwidth and storage efficiency, with two extremal cases: 1) the *minimum-storage regenerating* (MSR) point where each node stores the minimum amount of data, and 2) the *minimum-bandwidth regenerating* (MBR) point where the storage efficiency is sacrificed for achieving the minimum repair bandwidth.

In practice, minimum repair bandwidth does not always result in minimum regeneration time, especially in heterogeneous networks where link capacities vary in a wide range. The heterogeneous networks are commonly deployed for distributed storage systems. For example, within a data center, servers are usually placed in racks, and the servers in the same rack may enjoy a much larger bandwidth than those located in different racks [4]. Meanwhile, the available bandwidths among servers are distinct because of different background traffic, even if the link capacities are the same [5]. The difference of link bandwidths becomes even larger when using multiple geo-distributed data centers to safeguard users' data from the failure of an entire data center, which is a conventional practice for large companies, *e.g.* Google.

Y. Wang is with the School of Software, East China Jiao Tong University, Nanchang, China. The main work was done when she was with the School of Computer Science, Fudan University, email: ywangc11@fudan.edu.cn

X. Yin is with Department of Computer Science, University of Calgary, Canada.

D. Wei and X. Wang are with the School of Computer Science, Fudan University.

Y. He is with the School of Information Science and Engineering, Huaqiao University, Xiamen, China.

This paper was presented in part at IEEE INFOCOM 2014.

In these networks, the regeneration time depends not only on the repair bandwidth, but also on the bandwidths of bottleneck links between providers and the newcomer. Thus, in this work, we will also consider nonuniform repair traffic in order for the amount of transmitted repair data to match the available bandwidth over heterogeneous links.

There are, in general, two approaches for accelerating the regeneration process in heterogeneous networks. The first is to drop the assumption of uniform repair traffic. Instead, the repair traffic would better be dynamically determined according to the end-to-end available bandwidth from each provider to the newcomer. The idea is to let providers with larger end-to-end available bandwidths transmit more repair data for reducing in turn the amount of data transmitted through bottleneck links. This approach requires a new form of restriction on the repair traffic to maintain the MDS property. For example, Shah *et al.* [6] developed a regenerating scheme that supports non-uniform repair traffic with two newly introduced parameters β_{max} and γ . While the repair traffic can be dynamically determined for each provider in each regeneration process, it requires that the provider transmits at most an amount of β_{max} repair data and the total amount of repair data from all the providers must be no less than γ , so that the MDS property can be preserved.

The second approach is to utilize the inter-provider links to bypass bottleneck links between providers and the newcomer. This idea was first proposed by Li *et al.* [7]. They designed a *tree-structured* regeneration scheme (called *RCTREE*), where the tree has the newcomer as the root and has providers as intermediate and leaf nodes, and the repair data reach the newcomer along the branches of the tree. In addition, they allowed the repair data to be encoded at the intermediate nodes of the tree to further reduce the regeneration time. However, because of insufficient amount of repair data transmitted, *RCTREE* cannot maintain the MDS property.

In this paper, we study the problem of minimizing regeneration time in heterogeneous networks while maintaining the MDS property. Our contributions include two parts as described below.

First, we propose a regenerating scheme supporting non-uniform repair traffic. Compared with previous studies, our scheme has the advantage of choosing the repair traffic according to the set of largest available bandwidths while maintaining the MDS property, and thus achieves the best possible regeneration time among the schemes those are based on dynamic determination of repair traffic. We introduce the *feasible region* to generalize the restrictions on the repair traffic for the MDS property. For the MSR case, we study the structure and the uniqueness of maximal feasible regions and then find their optimal solution. For the non-MSR case, we construct a heuristic feasible region with which the repair traffic can be dynamically determined by solving a linear programming problem for each round of repair. We refer to this regeneration scheme as flexible regeneration (FR).

Second, we reconsider the tree-structured regeneration approach with regenerating codes, because the previous tree-structured regeneration scheme [7] cannot preserve the MDS property in the repair process. We develop the information

flow graph method from [3]. Using this method, we derive the minimum amount of data to be transmitted on each link in a given regeneration tree, and formulate the problem of building an optimal regeneration tree to minimize the regeneration time. Unfortunately, we find this problem NP-complete, mainly because the information flow on each link exhibits a correlation with the number of providers using this link. We thus propose a heuristic algorithm, called Tree-structured Regeneration (TR), to find a near-optimal regeneration tree. Furthermore, we propose a Flexible Tree-structured Regeneration (FTR) scheme by combining TR with FR.

These ideas can be illustrated with the example shown in Fig. 1. Consider the overlay network shown in Fig. 1(a), where v_0 is the newcomer, and v_1, v_2, v_3, v_4 are the $d = 4$ providers. The bandwidths are labeled on the links, which range from 5Mbps to 70Mbps. We assume that the redundancy is coded as an $(n = 5, k = 2)$ -MDS code, such that any 2 out of 5 storage nodes are able to reconstruct the file. Suppose that the size of the original file is equal to $M = 480\text{Mb}$, and each storage node stores $\alpha = M/k = 240\text{Mb}$.

Fig. 1(b) shows the conventional regeneration scheme (STAR)[3] which uses the star-structured topology: v_0 receives data directly from the four providers. In order to regenerate the lost data at v_0 using the regenerating code, an amount of $\beta = \frac{M}{k(d-k+1)} = 80\text{Mb}$ data need to be downloaded from each provider. Thus, the regeneration time of STAR is determined by the slowest transmission which takes $\frac{\beta}{10\text{Mbps}} = 8$ seconds. Fig. 1(c) shows the STAR-based flexible regeneration (FR) scheme, where each provider is allowed to generate a different amount of coded data. We find that the MDS property can be maintained if providers v_1, v_2, v_3, v_4 generate and send $\beta_1 = 150\text{Mb}$, $\beta_2 = 150\text{Mb}$, $\beta_3 = 60\text{Mb}$, $\beta_4 = 30\text{Mb}$, respectively. As a result, the regeneration time of FR is reduced to $\max\{\frac{\beta_1}{70\text{Mbps}}, \frac{\beta_2}{50\text{Mbps}}, \frac{\beta_3}{20\text{Mbps}}, \frac{\beta_4}{10\text{Mbps}}\} = 3$ seconds.

The regeneration trees used by TR and FTR are shown in Fig. 1(d) and (e), respectively. They coincide with the same tree in this example. In TR, each provider generates $\beta = 80\text{Mb}$ coded data with its local storage. According to our analysis, the minimum amount of data transferred on edge (v_1, v_0) is at least 2β . Thus, it costs TR $\max\{\frac{2\beta}{70\text{Mbps}}, \frac{\beta}{50\text{Mbps}}, \frac{\beta}{20\text{Mbps}}, \frac{\beta}{35\text{Mbps}}\} = 4$ seconds to accomplish the regeneration. With FTR, each provider may generate different amount of repair data. The solution given by our FTR algorithm is that providers v_1, v_2, v_3, v_4 generate $\beta_1 = 133.33\text{Mb}$, $\beta_2 = 133.33\text{Mb}$, $\beta_3 = 53.33\text{Mb}$, $\beta_4 = 53.33\text{Mb}$, respectively. The amount of data flow transmitted on the link (v_1, v_0) is $\beta_1 + \beta_4 = 186.67\text{Mb}$, and the regeneration time is $\max\{\frac{\beta_1 + \beta_4}{70\text{Mbps}}, \frac{\beta_2}{50\text{Mbps}}, \frac{\beta_3}{20\text{Mbps}}, \frac{\beta_4}{35\text{Mbps}}\} = 2.67$ seconds.

To evaluate the performances of our schemes, we implement these regeneration schemes and carry out simulations with practical link capacities. Simulation results show that, depending on heterogeneous link capacities, our proposed schemes can reduce the regeneration time by 10% ~ 90%, compared with the conventional STAR-structured regeneration scheme.

The remainder of the paper is organized as follows. In Section II, we formulate the process of data regeneration in distributed storage systems and introduce the random linear coding and information flow graph, which is the theoretical

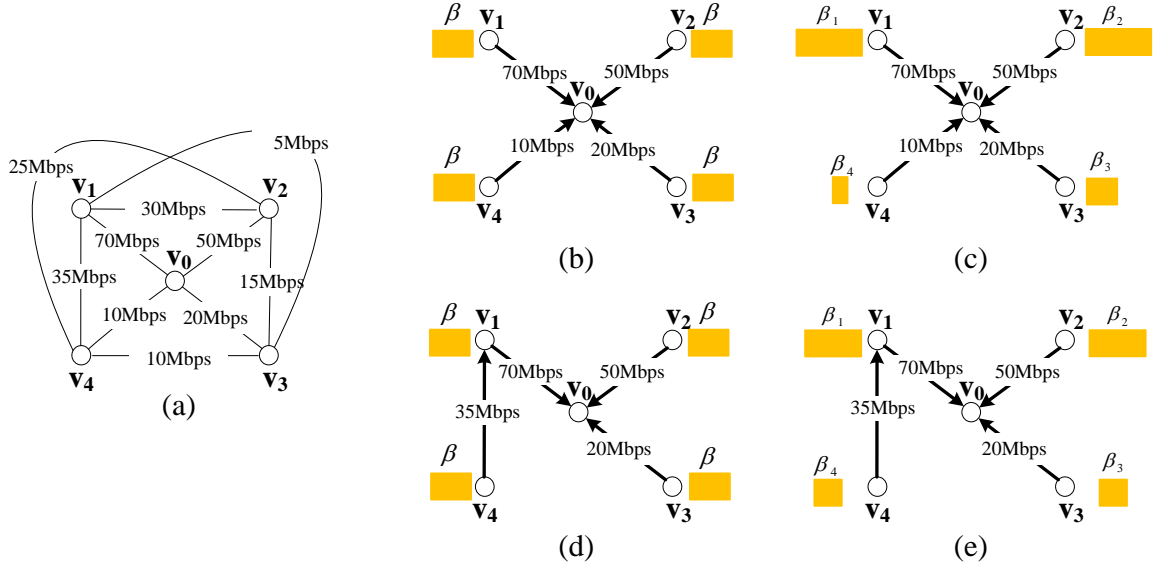


Fig. 1. Examples for tree regeneration schemes: STAR, FR, TR and FTR. The parameters are $n = 5$, $d = 4$, $k = 2$, $M = 480\text{Mb}$, $\alpha = M/k = 240\text{Mb}$, $\beta = \frac{\alpha}{d-k+1} = 80\text{Mb}$. The lengths of the blocks indicate the amount of repair data generated from each provider. The regeneration time of STAR, FR, TR and FTR is 8s, 4s, 3s and 2.67s, respectively.

tool applied in the analysis. In Section III, we study the flexible regeneration. In Section IV, we reconsider the tree-structured regeneration scheme by analyzing the corresponding information flow graph. In Section V, we propose a flexible scheme based on tree-structured regeneration. In Section VI, we evaluate the performance of our three proposed regeneration schemes. Finally, we introduce the related works in Section VII, and conclude the paper by Section VIII.

II. PROBLEM FORMULATION

From this section, we expand the unit from 'bit' to 'block' that comprises a specific length of bits for describing the variables M , α , β , and β_i .

Assume that a file is divided into M blocks and encoded into $n\alpha$ blocks of equal length. The coded blocks are disseminated to n storage nodes, with each node holding α blocks. The reliability requirement is formalized as the MDS property, which requires that the file can be reconstructed by accessing any k storage nodes. After a storage node fails, the newcomer accesses d providers to regenerate α blocks. With regenerating codes, each provider generates β coded blocks by encoding the local α blocks and directly transmits them to the newcomer.

We use a complete graph $G(V, E)$ to represent the overlay network consisting of the d providers and the newcomer [7]. The d provider-to-newcomer flows form a star topology centered at the newcomer. For any two nodes $u, v \in V$, let $c(u, v)$ denote the link capacity from u to v . For a specific regeneration process, let $f(u, v)$ denote the number of blocks transmitted over the link (u, v) . We assume that the coding operations are streamlined with the data transmission, which dominates the regeneration time. Notice that in real-world overlay networks, the end-to-end links usually have different capacities, which may vary in one or two orders of magnitude

[8]. The regeneration time can be simply represented as

$$\max \left\{ \frac{f(u, v)}{c(u, v)} \mid (u, v) \in E \right\}$$

The challenge is how to minimize the regeneration time without violating the MDS property. The regeneration process based on the star topology may suffer from a bottleneck caused by the lowest link capacity between the providers and the newcomer. In general, we can utilize two approaches to accelerate the regeneration process in heterogeneous networks. The first is to generate a different amount of repair data from each provider according to its outgoing link capacity. The second is to utilize the links between providers to by-pass links of low capacities.

Therefore, a regeneration scheme can be decomposed into two parts: 1) an algorithm that flexibly determines the repair flows $f(u, v)$, $(u, v) \in E$, while assuring that the newcomer obtains enough information to regenerate the desired data; 2) a set of codes indicating how to encode the blocks and how to reconstruct the file. For the latter part, we employ the random linear network codes, with which we are able to utilize the information flow graph technique [3] to simplify the problem.

A. Random linear network coding and information flow graph

Dimakis *et al.* [3] proposed the information flow graph technique to analyze the minimum repair bandwidth for maintaining the MDS property. Specifically, they construct the information flow graph in the following way. For each storage node u , create two nodes u^{in} (in-node) and u^{out} (out-node) and a link of capacity α from u^{in} to u^{out} . Create a source node s and for each initial storage node u , add a link from s to u^{in} with infinite capacity. For a storage node v regenerated by accessing d providers, add d links from the out-nodes of the d providers to the in-node of newcomer v , each with capacity β .

For the purpose of proposal in this paper, we assume various link capacities β_i between the providers and the newcomer. If a data collector (DC) connects to k storage nodes to reconstruct the file, add k links of infinite capacity from the out-nodes of these storage nodes to the data collector.

In the distribution process, the original file is divided into M blocks $a_j (j = 1, 2, \dots, M)$ of equal length, and is encoded into $n\alpha$ blocks $b_i (i = 1, 2, \dots, n\alpha)$. These coded blocks are then evenly distributed to n storage nodes. With linear codes, each coded block b_i is a linear combination of blocks a_1, a_2, \dots, a_M , i.e.,

$$\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n\alpha} \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,M} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n\alpha,1} & c_{n\alpha,2} & \cdots & c_{n\alpha,M} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix}$$

For the MDS property, we have to carefully choose the combination coefficients $c_{i,j}$ so that the original blocks a_j can be reconstructed from any k storage nodes. A possible encoding method is to utilize Reed-Solomon codes [9], e.g., setting the generator matrix $[c_{i,j}]$ to be a Vandermonde matrix. For ease of implementation, the combination coefficients $c_{i,1}, c_{i,2}, \dots, c_{i,M}$ for any specific coded block $b_i = \sum_{j=1}^M c_{i,j} a_j$ are transmitted and stored together with block b_i . For simplicity, we use $\vec{c} = (c_1, c_2, \dots, c_M)$ to denote the row vector of combination coefficients, which is also called the coding vector of block b .

In the regeneration process, we employ a random linear network coding scheme as the regenerating code to facilitate the regeneration of coded blocks at the newcomer. Without loss of generality, each provider v_i generates and transmits β_i coded blocks $b_{i,1}^{(r)}, b_{i,2}^{(r)}, \dots, b_{i,\beta_i}^{(r)}$ as random linear combinations of its local α blocks $b_{i,1}, b_{i,2}, \dots, b_{i,\alpha}$ as follows

$$\begin{bmatrix} b_{i,1}^{(r)} \\ b_{i,2}^{(r)} \\ \vdots \\ b_{i,\beta_i}^{(r)} \end{bmatrix} = \begin{bmatrix} c_{1,1}^{(r)} & c_{1,2}^{(r)} & \cdots & c_{1,\alpha}^{(r)} \\ c_{2,1}^{(r)} & c_{2,2}^{(r)} & \cdots & c_{2,\alpha}^{(r)} \\ \vdots & \vdots & \ddots & \vdots \\ c_{\beta_i,1}^{(r)} & c_{\beta_i,2}^{(r)} & \cdots & c_{\beta_i,\alpha}^{(r)} \end{bmatrix} \begin{bmatrix} b_{i,1} \\ b_{i,2} \\ \vdots \\ b_{i,\alpha} \end{bmatrix}$$

where $i = 1, 2, \dots, d$, and the coefficients $c_{i,j}^{(r)}$ are randomly chosen from the finite field of the linear regenerating code such that the coefficient matrix $[c_{i,j}^{(r)}]$ has rank β_i .

To complete the regeneration process, the newcomer receives $\gamma = \sum_{i=1}^d \beta_i$ blocks and generates its own α blocks by

$$\begin{bmatrix} b'_1 \\ b'_2 \\ \vdots \\ b'_\alpha \end{bmatrix} = \begin{bmatrix} c'_{1,1} & c'_{1,2} & \cdots & c'_{1,\gamma} \\ c'_{2,1} & c'_{2,2} & \cdots & c'_{2,\gamma} \\ \vdots & \vdots & \ddots & \vdots \\ c'_{\alpha,1} & c'_{\alpha,2} & \cdots & c'_{\alpha,\gamma} \end{bmatrix} \begin{bmatrix} b_{1,1}^{(r)} \\ b_{1,\beta_1}^{(r)} \\ b_{2,1}^{(r)} \\ b_{2,\beta_2}^{(r)} \\ \vdots \\ b_{d,\beta_d}^{(r)} \end{bmatrix}$$

where the coefficient matrix $[c'_{i,j}]$ is obtained from the coefficient matrices $[c_{i,j}^{(r)}]$ for the linear regenerating code. It should be noted that the coding vectors of the blocks b'_i are also similarly generated and stored at the newcomer.

In the reconstruction process, the data collector collects $k\alpha$ blocks $b''_1, b''_2, \dots, b''_{k\alpha}$ from k storage nodes, whose coding vectors form a $k\alpha$ -by- M matrix $[c''_{i,j}]$. As long as the matrix has rank M , we can reconstruct the original file from these coded blocks (as required by the MDS property [1]) by solving the following linear equation:

$$\begin{bmatrix} b''_1 \\ b''_2 \\ \vdots \\ b''_{k\alpha} \end{bmatrix} = \begin{bmatrix} c''_{1,1} & c''_{1,2} & \cdots & c''_{1,M} \\ c''_{2,1} & c''_{2,2} & \cdots & c''_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ c''_{k\alpha,1} & c''_{k\alpha,2} & \cdots & c''_{k\alpha,M} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix}$$

With the information flow graph, Dimakis *et al.* proved the following result [3]:

Lemma 1: In the information flow graph constructed according to the repair history of a distributed storage system, if the max-flow from s to a data collector is no less than the file size M , then with random linear codes over field \mathbb{F} , the data collector can recover the file with probability arbitrarily close to 1 as $|\mathbb{F}| \rightarrow \infty$.

III. STAR-STRUCTURED REGENERATION WITH FLEXIBLE REPAIR TRAFFIC

We first consider the approach of utilizing flexible non-uniform repair traffic to accelerate the regeneration process with heterogeneous link capacities. Generalizing the set of repair traffic that preserves the MDS property as the “feasible region”, we characterize the structure of maximal feasible regions and derive the flexible regeneration scheme whose feasible region subsumes the feasible region of previous studies.

Specifically, for a regeneration process, let β_i denote the repair traffic, i.e., the number of blocks transmitted from the i -th provider to the newcomer, and let $\beta = (\beta_1, \beta_2, \dots, \beta_d)$ denote the repair bandwidth in terms of the vector of repair traffic. For each round of repair, if we have multiple choices of β , then we say that ‘flexible repair traffic is supported’. Let \mathcal{D} denote the set of possible choices of β . We call $\mathcal{D} \subset \mathcal{R}^d$ a feasible region, if the MDS property is maintained as long as β is chosen from \mathcal{D} .

With a given feasible region, minimizing the regeneration time in each round of repair is equivalent to solving:

$$\min_{\beta \in \mathcal{D}} \max_{i=1, \dots, d} \frac{\beta_i}{c(v_i, v_0)} \quad (1)$$

where v_i is the i -th provider and v_0 is the newcomer. The link capacities $c(v_i, v_0)$ may be simply written as c_i .

According to the analysis of feasible regions in the following subsection, we can see that a maximal feasible region is actually a convex polytope, which makes problem (1) a linear programming problem and solvable in polynomial time.

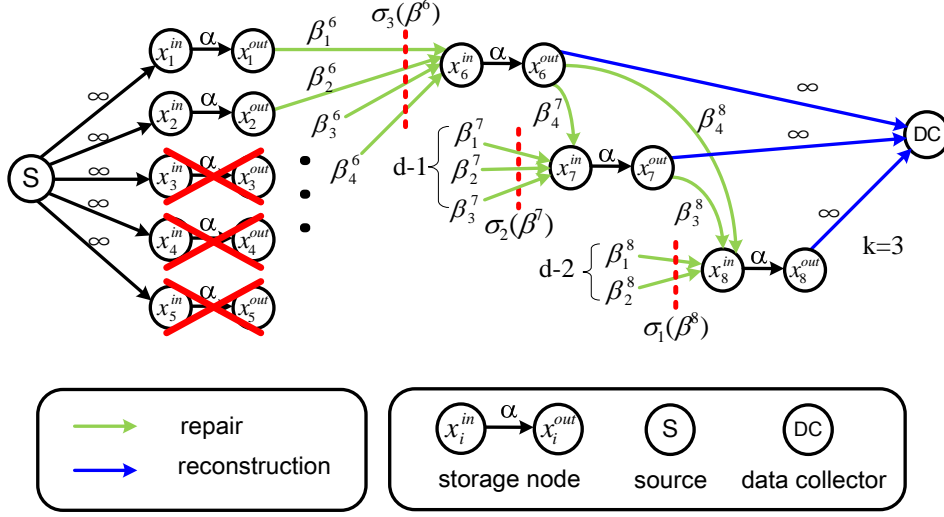


Fig. 2. An example of the information flow graph when $k = 3, d = 4$, where β^i denotes the repair bandwidth in the repair when the node x_{i+n}^i acts as the newcomer. Without loss of generality, we suppose that the elements of β^i are sorted such that $\beta_1^i \leq \beta_2^i \leq \beta_3^i \leq \beta_4^i, i = 0, 1, 2$, and let $\sigma_j(\beta^i) = \sum_{l=1}^{d-k+j} \beta_l^i$. We can see that the min-cut equals $\sum_{j=1}^k \min\{\sigma_j(\beta^{j-1}), \alpha\}$ in this figure.

A. Structure of a maximal feasible region

In the information flow graph, the capacities of links entering a newcomer's in-node represent the amount of information downloaded from each provider. Therefore, with flexible repair traffic, we set the corresponding link capacities as $\beta_i, i = 1, \dots, d$, instead of β . With this minor difference from the conventional information flow graph, it can be seen that Lemma 1 still holds.

In order to convert the MDS property into constraints on the feasible region, we analyze the min-cuts of the information flow graph:

Lemma 2: If in each round of repair, the newcomer accesses d providers with the repair bandwidth $\beta \in \mathcal{D}$. Then in the corresponding information flow graph, the min-cut between s and any data collector DC satisfies

$$\text{min-cut}(s, \text{DC}) \geq \sum_{j=1}^k \min \left\{ \min_{\beta \in \mathcal{D}} \sigma_j(\beta), \alpha \right\} \quad (2)$$

Here $\sigma_j(\beta)$ is defined as the sum of the $d - k + j$ smallest numbers of $\beta_1, \beta_2, \dots, \beta_d$. This bound can be matched with equality.

Proof: Let (U, \bar{U}) denote a cut separating DC and s such that $\text{DC} \in U$ and $s \in \bar{U}$, respectively. Then U must contain at least k storage nodes. Consider each of the first k nodes of U in the topological sorting. If it is an input node, the set of repair links will be included in the cut. If it is an output node, the storage link of capacity α will be included. As illustrated in Fig. 2, it can be verified that the equality of (2) can be achieved when there are k cascading repairs, each of which includes the newcomers in previous rounds as helper nodes. ■

As the bound of the min-cut (2) can be matched with equality, we use $MC(\mathcal{D}, \alpha)$ to represent the minimum min-cut of all possible information flow graphs with a given region \mathcal{D} and the storage per node α . If we want to ensure that any

k storage nodes suffice to reconstruct the original file, while allowing a newcomer to connect to any set of d providers, then the following condition must be satisfied:

$$MC(\mathcal{D}, \alpha) = \sum_{j=1}^k \min \left\{ \min_{\beta \in \mathcal{D}} \sigma_j(\beta), \alpha \right\} \geq M. \quad (3)$$

We refer to this condition as the *min-cut condition*. A set $\mathcal{D} \subset \mathcal{R}^d$ is a feasible region if and only if it satisfies the min-cut condition.

Larger feasible region means more flexibility in finding the suitable repair bandwidth. Therefore, we only need to consider the maximal feasible regions. The following theorem shows that a maximal feasible region can be described by a k -tuple (x_1, x_2, \dots, x_k) .

Theorem 1: A maximal feasible region \mathcal{D} can be written in the following form:

$$\mathcal{D} = \{\beta | \sigma_j(\beta) \geq x_j, \quad j = 1, \dots, k\},$$

where $0 \leq x_1 \leq \dots \leq x_k \leq \alpha$ and $\sum_{j=1}^k x_j \geq M$.

Proof: Let \mathcal{D}' be a feasible region satisfying the min-cut condition in (3). As the sequence $\min_{\beta \in \mathcal{D}'} \sigma_j(\beta)$ is non-decreasing for $j = 1, \dots, k$, we let i be the largest integer such that $\min_{\beta \in \mathcal{D}'} \sigma_{k-i+1}(\beta) \geq \alpha$. Thus the min-cut of an information flow graph under \mathcal{D}' is

$$\sum_{j=1}^{k-i} \min_{\beta \in \mathcal{D}'} \sigma_j(\beta) + i\alpha \geq M.$$

If \mathcal{D}' cannot be rewritten in the required form, we construct another feasible region $\mathcal{D}'' \supset \mathcal{D}'$ as

$$\mathcal{D}'' = \{\beta | \sigma_j(\beta) \geq x_j, j = 1, \dots, k\}$$

where

$$x_j = \begin{cases} \min_{\beta \in \mathcal{D}'} \sigma_j(\beta), & j = 1, \dots, k-i, \\ \alpha, & j = k-i+1, \dots, k. \end{cases}$$

Since \mathcal{D}' satisfies the min-cut condition, we can deduce that $0 \leq x_1 \leq \dots \leq x_k \leq \alpha$ and $\sum_{j=1}^k x_j \geq M$. Thus, \mathcal{D}'' is of the required form. To complete the proof, it is sufficient to show that \mathcal{D}'' is maximal and satisfies the min-cut condition.

From the construction, we have $\mathcal{D}' \subset \mathcal{D}''$, since for any $\beta \in \mathcal{D}'$ it must hold that $\sigma_j(\beta) \geq x_j$. As \mathcal{D}' is a maximal feasible region, we conclude \mathcal{D}'' is maximal. We can verify that \mathcal{D}'' also satisfies the min-cut condition:

$$\begin{aligned} MC(\mathcal{D}'', \alpha) &= \sum_{j=1}^k \min \left\{ \min_{\beta \in \mathcal{D}''} \sigma_j(\beta), \alpha \right\} \\ &= \sum_{j=1}^{k-i} \min_{\beta \in \mathcal{D}''} \sigma_j(\beta) + i\alpha \\ &\geq \sum_{j=1}^{k-i} x_j + i\alpha \\ &= \sum_{j=1}^{k-i} \min_{\beta \in \mathcal{D}'} \sigma_j(\beta) + i\alpha \geq M \end{aligned}$$

B. The optimal feasible region for $\alpha = M/k$

For the MSR case where the minimum storage $\alpha = M/k$, we find that there exists one maximum feasible region, which minimizes the regeneration time for any link capacity settings.

Theorem 2: For the case of $\alpha = M/k$, any feasible region is a subset of the maximum feasible region

$$\mathcal{D}^* = \{\beta | \sigma_1(\beta) \geq M/k\}$$

Proof: First, it can be seen that for any set \mathcal{D}

$$\min_{\beta \in \mathcal{D}} \sigma_1(\beta) \leq \min_{\beta \in \mathcal{D}} \sigma_2(\beta) \leq \dots \leq \min_{\beta \in \mathcal{D}} \sigma_k(\beta)$$

because for any $j = 1, 2, \dots, k-1$

$$\min_{\beta \in \mathcal{D}} \sigma_{j+1}(\beta) = \sigma_{j+1}(\beta') \geq \sigma_j(\beta') \geq \min_{\beta \in \mathcal{D}} \sigma_j(\beta),$$

where β' is the repair bandwidth that minimizes $\sigma_{j+1}(\beta')$. Thus, we have

$$MC(\mathcal{D}^*, \alpha) \geq k \min_{\beta \in \mathcal{D}^*} \sigma_1(\beta) \geq M$$

which means that \mathcal{D}^* is indeed a feasible region.

On the other hand, for any feasible region \mathcal{D} and any $\beta \in \mathcal{D}$, if $\sigma_1(\beta) < M/k$, then

$$MC(\mathcal{D}, \alpha) < M/k + (k-1)\alpha = M$$

which contradicts the min-cut condition in (3). Thus, we must have $\sigma_1(\beta) \geq M/k$ and hence $\beta \in \mathcal{D}^*$. ■

As a conclusion, when using the MSR codes, we can set the feasible region to be \mathcal{D}^* and in each round of repair, we determine the repair bandwidth β by solving the optimization problem (1), which is reduced to:

$$\begin{aligned} \min_{\beta \in \mathcal{D}^*} \quad & \max_{i=1, \dots, d} \frac{\beta_i}{c_i} \\ \text{subject to:} \quad & \sigma_1(\beta) \geq M/k \end{aligned} \quad (4)$$

where c_i is the link capacity from the i -th provider v_i to the newcomer v_0 .

Without loss of generality, we may assume $c_1 \leq c_2 \leq \dots \leq c_d$. Then the optimal solution β^* to problem (4) can be solved explicitly as follows

$$\beta_j^* = \begin{cases} \frac{c_j M}{k \sum_{i=1}^{d-k+1} c_i} & \text{if } 1 \leq j \leq d-k+1 \\ \frac{c_{d-k+1} M}{k \sum_{i=1}^{d-k+1} c_i} & \text{if } d-k+1 < j \leq d \end{cases}$$

C. The feasible region for $\alpha > M/k$

For the case of $\alpha > M/k$, we find out that there does not exist a maximum feasible region for any $\alpha > M/k$ and $k \geq 3$ (Please refer to appendix for a detailed proof). In other words, the optimal feasible region depends on the link capacities. While regenerating data for a new node, we still do not know the link capacities of helper nodes that will join when future failures occur, so we cannot determine the conditions for the optimal solution. Here we use an example to explain this statement.

Example 1: Set $n = 5$, $k = 3$, $d = 4$, $M = 12$, and $\alpha = 6$, and consider the following two feasible regions:

$$\begin{aligned} \mathcal{D}_1 &= \{\beta | \sigma_1(\beta) \geq 1, \sigma_2(\beta) \geq 5, \sigma_3(\beta) \geq 6\}, \\ \mathcal{D}_2 &= \{\beta | \sigma_1(\beta) \geq 2, \sigma_2(\beta) \geq 4, \sigma_3(\beta) \geq 6\}. \end{aligned}$$

We find two different repair bandwidths, $\beta_1 = (0, 1, 4, 4) \in \mathcal{D}_1 \setminus \mathcal{D}_2$ and $\beta_2 = (0, 2, 2, 2) \in \mathcal{D}_2 \setminus \mathcal{D}_1$. If the corresponding link capacities c_i are given as $(1, 1, 4, 4)$, the regeneration times for β_1 and β_2 are 1 second and 2 seconds, respectively. Under this capacity setting, \mathcal{D}_1 is a better solution. However, with another setting of link capacities $(1, 2, 2, 2)$, the regeneration times are then changed to 2 seconds and 1 second, respectively. Feasible region \mathcal{D}_2 outperforms \mathcal{D}_1 in this setting. According to our previous analysis, $\mathcal{D}_1, \mathcal{D}_2$ are both maximal feasible regions and there does not exist a feasible region including both of them. Thus it is unable to minimize the regeneration times for both the above capacity settings simultaneously.

From this example, we can conclude that, in order to minimize the regeneration time, knowledge on the provider-to-newcomer link capacities for the next rounds of repairs is necessary, which is impractical in real-world distributed storage systems. Therefore, we propose a heuristic feasible region instead:

$$\mathcal{D}^* = \{\beta | \sigma_j(\beta) \geq \min\{(d-k+j)\beta, \alpha\}, j = 1, \dots, k\}$$

where β is the amount of data downloaded from each provider in the conventional regenerating scheme, which can be calculated according to the optimal tradeoff between storage α and the total repair bandwidth $\sum_{i=1}^d \beta_i$ from β [3]. It can be seen that the repair traffic of conventional regenerating scheme $\beta = (\beta, \dots, \beta)$ belongs to \mathcal{D}^* . Therefore, with this feasible region, the regeneration time will never be worse than the conventional regenerating scheme.

With the feasible region \mathcal{D}^* , we can determine the amount of data to be downloaded from each provider in each round of

repair by solving the linear programming problem (1). In case that the solution $\beta = (\beta_1, \dots, \beta_d)$ takes on fractional values, its components can be rounded up to their nearest integers. Note that we may choose large M to make the rounding error negligible.

IV. TREE-STRUCTURED REGENERATION WITH CONSTANT REPAIR TRAFFIC

Li *et al.* [7] first proposed a tree-structured regeneration scheme which transmits the regeneration traffic along a carefully selected tree spanning all the providers. However, as shown in the appendix, this method cannot maintain the MDS property. In this section, we reconsider the problem of optimizing the regeneration time for tree-structured regeneration and maintaining the MDS property featured by entire distributed storage systems.

A tree-structured regeneration solution has two parts: the regeneration tree $T \subset E$ and the number of blocks $f(u, v)$ transmitted on each link of the tree. In the following subsections, we first study the minimum $f(u, v)$ in a given regeneration tree to preserve the MDS property. Thereafter, we show that the problem of building an optimal regeneration tree is NP-hard. Finally, we conclude this section with a heuristic algorithm for constructing a regeneration tree.

A. The minimum $f(u, v)$ for a given regeneration tree

As the conventional information flow graph is originally constructed for the star topology, we first generalize it to the tree topology for regeneration.

To construct the information flow graph for tree-structured regeneration, instead of simply adding d links from the providers' out-nodes to the newcomer's in-node, we rather add links according to the regeneration tree T and the number of blocks $f(u, v)$ transmitted on each link of the tree. During the regeneration, if provider u transmits $f(u, w)$ blocks to provider w , we add a link from u^{out} to w^{out} with capacity $f(u, w)$, whereas if provider u transmits $f(u, v)$ blocks directly to the newcomer v , we add a link from u^{out} to v^{in} with capacity $f(u, v)$. For example, Fig. 3(a) shows a regeneration tree consisting of three providers u_1, u_2, u_3 and a newcomer v , and Fig. 3(b) presents the corresponding part of information flow graph.

With the generalized information flow graph, we are able to determine the minimum flow $f(u, v)$ on each link that ensures the MDS property.

Theorem 3: For a given regeneration tree T rooted at the newcomer, in order to preserve the MDS property, the minimum number of blocks transmitted on each link $(u, v) \in T$ is

$$\min\{m_u\beta, \alpha\}$$

where m_u is the number of nodes in the subtree rooted at u , and β is the number of blocks transmitted in the conventional regenerating scheme, which satisfies $\sum_{i=1}^k \min\{(d-i+1)\beta, \alpha\} = M$.

Proof: The key is to compute the min-cut of the generalized information flow graph. Let $[U, \bar{U}]$ denote a min-cut that

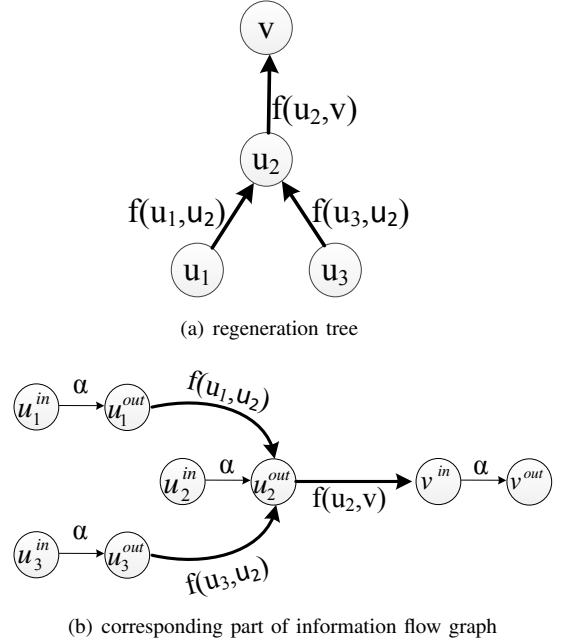


Fig. 3. An example of the part of the generalized information flow graph for a regeneration tree consisting of three providers u_1, u_2, u_3 (i.e., $d = 3$) and a newcomer v .

separates the data collector DC from the source, where U is the set of nodes containing DC. Then U must contain at least k "out" nodes. Label the k corresponding storage nodes as u_1, u_2, \dots, u_k in a topological order, such that u_i is a provider in regenerating u_j only if $i < j$. Fig. 4 demonstrates the min-cut in an information flow graph for a series of tree-structured regeneration processes for $d = 4$ and $k = 3$, where u_i is a provider with respect to u_j for $1 \leq i < j \leq 3$, and U contains all the in-nodes and out-nodes of u_1, u_2, u_3 .

We divide the cut links into k sets as follows: if $u_i^{in} \notin U$, then the i -th set contains only one link (u_i^{in}, u_i^{out}) ; otherwise, the i -th set contains all the cut-links introduced in the repair of node u_i . According to our scheme, a link (u, v) has flow rate $m_u\beta$ only if the subtree rooted at u has m_u nodes. For repairing u_i , as there are at most $i - 1$ providers in U , there are at least $d - i + 1$ providers in \bar{U} . Thus the total flow rate of the i -th set is no less than $\min\{(d - i + 1)\beta, \alpha\}$.

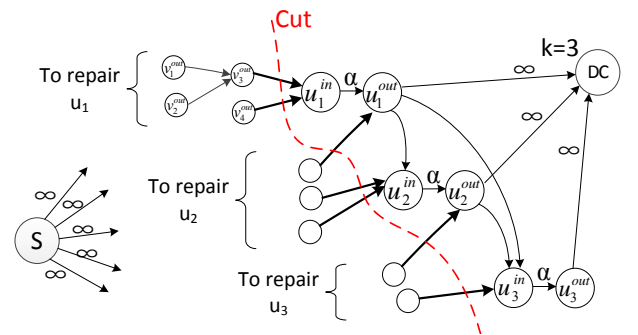


Fig. 4. An example of min-cut in an information flow graph in the tree-structured regeneration, where $d = 4$, and $k = 3$. The dashed line shows the min-cut and the bold links are the cut-links.

Sum up the volumes of the cut links. The total volume of the cut $[U, \bar{U}]$ must be no less than $\sum_{i=1}^k \min\{(d-i+1)\beta, \alpha\} = M$. Thus, if $f(u, v) \geq \min\{m_u\beta, \alpha\}$, DC can construct the file by assessing any k storage nodes.

To show that we cannot further reduce $f(u, v)$, we need to show that the min-cut with the minimum volume M is achievable. According to the information flow graph constructed in the proof, we can know that such a min-cut is achievable if we require each provider to provide β blocks, which means that the solution $\min\{m_u\beta, \alpha\}$ is optimal for a given regeneration tree. ■

B. Construction of the optimal regeneration tree

With the knowledge of how much information to be transmitted on each link, the remaining task is to build an optimal regeneration tree T to minimize the regeneration time. However, we find that the optimal regeneration tree (ORT) problem is NP-hard. To demonstrate this, we start with the formal definition of the ORT problem.

Definition 1: For a given overlay network $G(V, E)$ with link capacities $c(u, v)$, $(u, v) \in E$, the optimal regeneration tree problem is to find a spanning tree T such that the regeneration time $\max\{\frac{f(u, v)}{c(u, v)} | (u, v) \in T\}$ is minimized, where $f(u, v) = \min\{m_u\beta, \alpha\}$ and m_u is the number of nodes in the subtree rooted at $u \in V$.

To study the complexity of the ORT problem, it is equivalent to restating this optimization problem as a decision problem, which aims to determine whether the regeneration time of an optimal regeneration tree is no more than 1. The following theorem shows that this problem is NP-hard.

Theorem 4: The ORT problem is NP-hard.

Proof: We first show that ORT \in NP. Suppose we are given a graph $G = (V, E)$. The certificate we choose is the optimal regeneration tree T . The verification algorithm checks, for each edge $(u, v) \in T$, that $\frac{f(u, v)}{c(u, v)} \leq 1$. This verification can be performed in polynomial time.

We now prove that the ORT problem is NP-hard by reduction from the VERTEX-COVER problem, which is known to be NP-complete. In particular, given an undirected graph $G = (V, E)$ and an integer k , the VERTEX-COVER problem asks whether all edges can be “covered” by k nodes, where node $u \in V$ can cover edge $e \in E$ only if they are adjacent. For an instance of the VERTEX-COVER problem, we construct a regeneration scenario in an overlay network $G' = (V', E')$, such that the regeneration time is less than 1 if and only if G has a vertex cover of size k .

We construct G' in the following way. G' has four layers of nodes. The first layer has only one node that is the root t . The second layer has two nodes, node a and node b . Both a and b are connected to the root. The link capacity of edge (a, t) is $k+|E|+1$ and the link capacity of edge (b, t) is unlimited. The nodes in the third layer correspond to the vertices in graph G , all of them are connected to both a and b . The link capacity of each edge connected to a is unlimited, whereas the link capacity of each edge connected to b is 1. The nodes in the last layer correspond to the edges in graph G . Each node in the last layer is connected to two nodes in the third layer by the

corresponding edges in graph G . The edges, which connect the last layer nodes to the third layer nodes, each have an unlimited capacity. Links that are not mentioned in the construction are supposed to have zero capacity.

From the construction above, graph G' can be constructed from G in polynomial time. Fig. 5 shows an example of this reduction for the VERTEX-COVER problem with $k = 2$, where Fig. 5(a) is an instance of G , and Fig. 5(b) is the graph G' constructed from G .

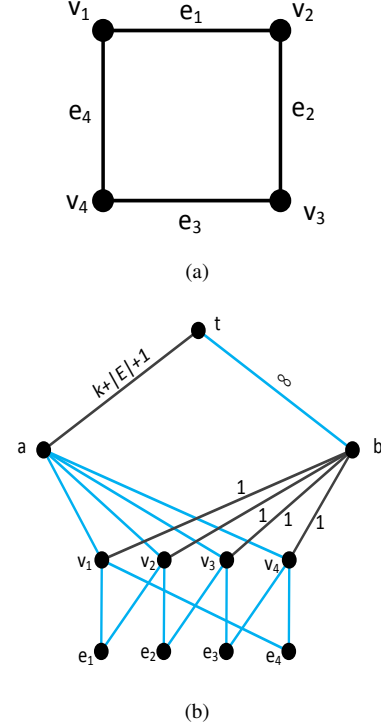


Fig. 5. Reduction of the VERTEX-COVER problem to the ORT problem for $k = 2$: (a) An undirected graph $G = (V, E)$; (b) The graph G' produced by the reduction procedure.

We next show that this transformation of G into G' is a reduction. First, suppose that G has a vertex cover set $V' \subseteq V$, where $|V'| = k$. We claim that we can find a tree whose regeneration time is no more than 1. This tree is constructed according to the vertex cover as follows. For each node e_i of the fourth layer, let its parent be v_j which covers the edge e_i in the vertex cover of G . For each node v_i of the third layer, if it belongs to the vertex cover V' , let its parent be a ; otherwise, let its parent be b . Finally, let the parent of a and b be the root. It can be verified that the regeneration time is exactly equal to 1.

Conversely, suppose that G' has a tree whose regeneration time is no more than 1. Then, we claim that the edges of G can be covered by no more than k nodes. Let $V' \subseteq V$ be the set of nodes that correspond to children of node a in the regeneration tree. First, V' is a vertex cover of G . If it is not the case, some nodes in the fourth layer will have to be connected to the root through b , causing some flow entering b larger than 1 and the regeneration time less than 1. Second, we show that $|V'| \leq k$. As all the $|E|$ nodes in the fourth layer must transmit their data to the newcomer through a , there will

be at least $|V'| + |E| + 1$ nodes transferring data from node a to root t . Because the capacity of link (a, t) is $k + |E| + 1$ and the regeneration time is no more than 1, we conclude that $|V'| \leq k$. ■

C. The heuristic algorithm for constructing the optimal regeneration tree

In this subsection, we propose a heuristic algorithm to solve the ORT problem since it is NP-hard as mentioned above. The algorithm is inspired by Prim's algorithm [10] for the maximum weighted spanning tree problem. We start from a tree containing only the newcomer as the root and iteratively add the remaining nodes to the tree until it spans all the providers. In each iteration, we try all possible positions for each remaining provider and choose the best position to add the corresponding provider into the regeneration tree. The details are shown in Algorithm 1, where v_0 represents the root for the newcomer, and (v', u') records the best positions.

Algorithm 1 Find a regeneration tree T for a given network $G(V, E)$.

- 1: Input: Network topology $G(V, E)$, link capacities $c(u, v)$, storage amount α , repair traffic β
- 2: Output: Regeneration tree T
- 3: $T \leftarrow v_0$
- 4: $A \leftarrow V - \{v_0\}$
- 5: **while** $A \neq \emptyset$ **do**
- 6: **for all** $v \in A$ **do**
- 7: **for all** $u \in T$ **do**
- 8: Compute the regeneration time $\max\{\frac{f(u, v)}{c(u, v)} \mid (u, v) \in T \cup \{(v, u)\}\}$ for tree $T \cup \{(v, u)\}$
- 9: If the regeneration time is better than previous choices, update $(v', u') \leftarrow (v, u)$
- 10: **end for**
- 11: **end for**
- 12: $T \leftarrow T \cup \{(v', u')\}$
- 13: $A \leftarrow A - \{v\}$
- 14: **end while**

The most time-consuming step is to test all possible positions for each provider, which has no more than $|V|^2$ choices. Each test takes a linear time of order $O(|V|)$ to compute the regeneration time. Thus, the algorithm runs in polynomial time of order $O(|V|^3)$.

V. TREE-STRUCTURED REGENERATION WITH FLEXIBLE END-TO-END TRAFFIC

In Sections III and IV, we have discussed two independent approaches to reduce the regeneration time: 1) allowing non-uniform end-to-end repair traffic; 2) allowing tree-structured regeneration topology. In this section, we propose a Flexible Tree-structured Regeneration (FTR) scheme, which combines the advantages of the two approaches to further reduce the regeneration time.

We present the logic flow of the proposed FTR scheme in three steps. First, we analyze the restrictions on the amount of repair data generated by each provider to maintain the MDS

property. Second, for a given regeneration tree, we calculate the optimal regeneration time based on the analysis in the first step. Finally, as we are able to determine which one of two trees results in a faster regeneration, we obtain a heuristic algorithm based on local searching.

A. A sufficient condition for the MDS property

Roughly stated, to maintain the MDS property under a flexible traffic strategy, if the providers connected to low-capacity links generate less coded blocks, then the providers connected to high-capacity links will have to generate more coded blocks. Let β_i denote the amount of repair traffic, i.e., the number of coded blocks generated by the i -th provider. Our first task is to analyze the explicit restrictions on β_i that ensures the MDS property.

As analyzed in Section IV, during the tree-structured regeneration, an intermediate node needs to re-encode the received blocks from its children only when the number of received blocks plus the number of blocks generated by itself is larger than α . In this case, the intermediate node transmits only α coded blocks. Thus, for a regeneration tree T , the number of coded blocks transmitted on each link $(u, v) \in T$ will be

$$f(u, v) = \min \left\{ \sum_{v_i \in S(u)} \beta_i, \alpha \right\}$$

where u denotes an intermediate node, v denotes the parent of u in the tree, $S(u)$ denotes the set of nodes in the subtree rooted at u , and β_i indicates the number of coded blocks generated at v_i and transmitted to u . The following theorem provides a sufficient condition for the MDS property.

Theorem 5: The MDS property is maintained if in each regeneration, we choose $\beta_i, i = 1, 2, \dots, d$, that satisfy

$$\sum_{l=1}^{d-k+j} \beta_{i_l} \geq \min\{(d-k+j)\beta, \alpha\} \quad \forall j = 1, 2, \dots, k$$

where β is again the number of coded blocks generated by a provider in the conventional regenerating scheme, and (i_1, i_2, \dots, i_d) is a permutation of $(1, 2, \dots, d)$ such that $\beta_{i_1} \leq \beta_{i_2} \leq \dots \leq \beta_{i_d}$.

Proof: We need to prove that any min-cut $[U, \bar{U}]$ ($DC \in U$) has volume of at least M .

As the link capacity from a storage node to the data collector DC is set to be infinity in the information flow graph, we only need to consider the case that U contains at least k storage nodes. Let v_1, v_2, \dots, v_k be the first k storage nodes of U in the topological order. For $j = 1, 2, \dots, k$, if the in-node of v_j is in the cut, all the links connected to v_j will also be included in the cut. However, if only the out-node of v_j is in the cut, the link from the in-node to the out-node of v_j that has capacity α will be included.

Following the way we determine the flow on the regeneration tree, for the j -th storage node v_j , the number of providers not in U will be at least $d-j+1$, and hence the total capacity of the cut links to v_j will be at least

$$\sum_{l=1}^{d-k+j} \beta_{i_l} \geq \min\{(d-k+j)\beta, \alpha\}$$

Therefore, the volume of the cut $[U, \bar{U}]$ will be no less than M , which ensures the MDS property. ■

B. Determination of the optimal regeneration time for a given tree

To support the non-uniform end-to-end repair traffic for a tree-structured regeneration, we introduce a parameter c_x to denote the end-to-end capacity from provider x to the newcomer v_0 , where x may not be directly connected to v_0 . Let t denote the regeneration time. For a given regeneration tree T , the maximum amount of data transmitted in t seconds through link (u, v) is $\sum_{x \in S(u)} t c_x$ if we do not perform encoding at the intermediate node u . However, Theorem 5 shows that if $\sum_{x \in S(u)} t c_x > \alpha$, we may encode the repair data at the intermediate node u and transmit only α blocks to node v . Therefore, we obtain the constraints on link capacities as follows

$$t c(u, v) \geq \min \left\{ \alpha, \sum_{x \in S(u)} t c_x \right\}, \quad \forall (u, v) \in T$$

Combining the constraints on the link capacities and the MDS property, we can write the optimal regeneration time as the following optimization problem:

$$\min t \quad (5)$$

subject to:

$$t c(u, v) \geq \min \left\{ \alpha, \sum_{x \in S(u)} t c_x \right\}, \quad \forall (u, v) \in T \quad (6)$$

$$t \sigma_1(\mathbf{c}) \geq (d - k + 1) \beta \quad (7)$$

where $\mathbf{c} = \{c_u \mid u \in V - v_0\}$, and $\sigma_1(\mathbf{c})$ is, just like in (2), defined as the sum of the $d - k + 1$ smallest components of \mathbf{c} .

C. The heuristic algorithm

Upon examining the linear programming (LP) problem in (5), we find that the optimal t is achieved by taking equality in the constraint (7) as

$$t = \frac{(d - k + 1) \beta}{\sigma_1(\mathbf{c})} \quad (8)$$

Substituting (8) into the problem (5), we can convert the objective from minimizing the regeneration time to maximizing $\sigma_1(\mathbf{c})$:

$$\max \sigma_1(\mathbf{c}) \quad (9)$$

subject to:

$$c(u, v) \geq \min \left\{ \frac{\alpha \sigma_1(\mathbf{c})}{(d - k + 1) \beta}, \sum_{x \in S(u)} c_x \right\}, \quad \forall (u, v) \in T \quad (10)$$

As there are an exponential number of different regeneration trees, we cannot enumerate all of them to find the optimal tree. Instead, we further study the structure of the LP problem (9) to perform a local search.

Note that the value $\frac{\alpha \sigma_1(\mathbf{c})}{(d - k + 1) \beta}$ is independent of links $(u, v) \in T$, and it is in fact a threshold on $\sum_{x \in S(u)} c_x$. We may decompose the constraint (10) into two parts by enumerating the number of links with capacity no less than the threshold. Denote this number by i . Then the feasible region given by (10) can be divided into $d + 1$ parts, with \mathbf{c} in the $(i + 1)$ -th part ($0 \leq i \leq d$) satisfying:

1) there are exactly i links, of which each has a capacity no less than the threshold;

2) for each of the rest $d - i$ links, its capacity $c(u, v) \geq \sum_{x \in S(u)} c_x$.

For each i , we run Algorithm 2 to find a candidate regeneration tree with i links having capacity no less than the threshold $\frac{\alpha \sigma_1(\mathbf{c})}{(d - k + 1) \beta}$ and finally pick up the best candidate as our regeneration tree.

Algorithm 2 A heuristic algorithm for Flexible Tree-structured Regeneration

- 1: Input: V, i, k , assuming $v_0 \in V$ is the newcomer.
 - 2: Output: A regeneration tree T that spans V , and a capacity allocation $\mathbf{c} = (c_u, u \in V - v_0)$ to maximize the sum of smallest m elements in \mathbf{c} under constraints P1 and P2
 - 3: Initialize: $T \leftarrow \emptyset, V' \leftarrow \{v_0\}$
 - 4: **for** $i' = 1$ to i **do**
 - 5: Among the links in the cut $[V', V - V']$, find the link (u, v) with the largest capacity (assuming $v \in V', u \notin V'$).
 - 6: $T \leftarrow T \cup \{(u, v)\}$
 - 7: $V' \leftarrow V' \cup \{u\}$
 - 8: **end for**
 - 9: Let $d' = d - i = |V - V'|$, $m = d - k + 1$
 - 10: **for each** $u \in V - V'$ **do**
 - 11: Let v' be the node maximizing $c(u, v)$ among nodes $v \in V'$
 - 12: $c_u \leftarrow c(u, v')$
 - 13: $T \leftarrow T \cup \{(u, v')\}$
 - 14: **end for**
 - 15: **repeat**
 - 16: Sort \mathbf{c} in ascending order $c_{u_1} \leq c_{u_2} \leq \dots c_{u_{d'}}$;
 - 17: For each $j > m$, set c_{u_j} to c_{u_m}
 - 18: **for each** $u \in V - V'$ **do**
 - 19: Let $(u, v) \in T$ be the link leaving u
 - 20: **for each** $v' \neq v$ **do**
 - 21: $T'' \leftarrow T - (u, v) + (u, v')$
 - 22: **if** \mathbf{c} is feasible in T'' and $\exists 1 \leq i \leq m, c_{u_i}$ can be increased **then**
 - 23: increase c_{u_i} to maximum possible under constraints P1 and P2
 - 24: $T \leftarrow T''$
 - 25: break;
 - 26: **end if**
 - 27: **end for**
 - 28: **end for**
 - 29: **until** T is not updated in the last loop
-

Algorithm 2 finds the candidate tree in two steps. First, through lines 3–8, we find a connected subtree containing i links so that the smallest link capacity of i links is maximized.

After the first step, V' is the set of i providers connected to the newcomer by the subtree. In the second step, we connect the rest $d' = d - i$ providers to the newcomer through a local search (lines 10–14), where we start from an initial regeneration tree, and during each iteration of the loop from line 15 to line 29, we check if we can get a better regeneration tree by a pivot operation that cuts off a subtree and connects it to some other node.

Note that for the case $i = 0$, lines 10–14 will set the initial regeneration tree as the star topology. Therefore, the regeneration tree returned by FTR is always no worse than the FR solution.

VI. EVALUATION

In this section, we present simulation results to verify the effectiveness of our proposed schemes: Flexible Regeneration (FR), Tree-structured Regeneration (TR), and Flexible Tree-structured Regeneration (FTR). Our most concern is the regeneration time, which is measured as the time that the newcomer spends on regenerating the coded blocks. In the evaluation of the regeneration time, we ignore the encoding time on each provider and the decoding time on the newcomer, because the encoding and decoding operations can be performed simultaneously during the transmission of repair data [7].

For default settings, we use the same experiment setup as [7], where redundant data is produced using an $(n = 20, k = 5)$ -MDS code. The original file size is set to be $M = 1\text{GB}$. The link capacities between the storage nodes obey the uniform distribution within the range $[10\text{Mbps}, 120\text{Mbps}]$, which is obtained from the measurement of real-world networks [8].

A. Effect of the number of providers d

The number of providers d is a key parameter for regeneration in distributed storage systems. In the STAR topology, the total repair bandwidth consumed in the regeneration process decreases as d grows [3]. In the case of one node failure, the theoretical optimal value of d is $n - 1$ for achieving a minimum repair bandwidth, although accessing a large number of providers will introduce extra communication overheads. On the other hand, all feasible values of d may appear in practice. In the evaluation, we vary d from $k + 1$ to $n - 1$, in order to find out how this factor affects the performance of each regeneration scheme.

We consider the MSR point, where each node stores $\alpha = M/k = 200\text{MB}$. Fig. 6 presents the simulation results on performance improvements of the FR, TR, and FTR schemes with respect to the STAR scheme, where all possible values of d are considered, and the uniform capacity distribution is within the range $[10\text{Mbps}, 120\text{Mbps}]$. Note that the regeneration schemes (STAR) proposed by Dimakis *et al.* in [3] is implemented as a benchmark. For convenience, the traditional STAR schemes based on uniform repair traffic are simply referred to as ‘STAR’ below.

In Fig. 6(a), we normalize the regeneration times of FR, TR and FTR by the regeneration time of STAR to show the relative improvement. In most cases, our schemes reduce the regeneration time by $50\% \sim 70\%$ compared with STAR.

We note that the regeneration times of both STAR and the three proposed regeneration schemes all decrease as d grows because of the reduction of total amount of repair data. Meanwhile, the regeneration times of our schemes reduce faster than that of STAR. This is because the star topology has a large chance to include a low capacitated provider-to-newcomer link when d is large. However, the bottleneck effect can be alleviated by FR and the tree topology.

An interesting observation is that FR outperforms TR for large d values, but on the contrary, TR outperforms FR when d is small. The reason is that, in order to bypass the bottleneck link with a tree topology, the intermediate provider nodes have to transmit more data. This effect can only be recovered by raising the minimum capacity for links connected to a big number of participating providers.

In Fig. 6(b), we examine the total repair bandwidth consumption for the FR, TR, and FTR schemes, where again the repair bandwidth is normalized by that for STAR. As a tradeoff, our schemes all sacrifice the repair bandwidth for reducing the regeneration time. However, it is not surprising that tree-structured regeneration has higher repair bandwidth consumption than STAR-structured regeneration. For example, in a regeneration tree the amount of repair data is counted twice if it is transmitted to the newcomer by two hops.

As a conclusion, FTR is always advantageous than both FR and TR in any case, which is promised by the design of FTR. When d is large, however, FR has a regeneration time almost as good as FTR, but enjoys a slightly smaller repair bandwidth.

B. Effect of the bandwidth variance

In order to show the impacts of network bandwidth variance on the regeneration time, we run simulations with 5 different link capacity distributions: $U_1[0.3, 120]\text{Mbps}$, $U_2[3, 120]\text{Mbps}$, $U_3[30, 120]\text{Mbps}$, $U_4[60, 120]\text{Mbps}$, $U_5[90, 120]\text{Mbps}$. Fig. 7 shows the results with the number of providers d fixed at 10. The performances of our schemes are better when the variance of network bandwidth is large. For uniform distribution $U_1[0.3, 120]\text{Mbps}$, FR, TR and FTR all achieve a reduction of about 90% in the regeneration time compared with the traditional STAR scheme. When the variance of network bandwidth becomes small, for example at $U_4[60, 120]\text{Mbps}$ and $U_5[90, 120]\text{Mbps}$, TR has the same regeneration time as STAR, but FTR still reduces the regeneration time by $10\% \sim 20\%$.

C. Effect of the storage capacity per node α

Our tests above focus mainly on the MSR point, which achieves the optimal storage efficiency. However, as shown by Dimakis *et al.* [3], the repair bandwidth can be reduced by storing more data on each storage node. To the other extreme, the MBR point achieves the minimum repair bandwidth.

To test the effects of the storage capacity per node on our schemes, we vary the values of α from the MSR point to the MBR point in the simulations. Fig. 8 shows the results on regeneration times and repair bandwidths for different α values. We find that the regeneration time in each of our

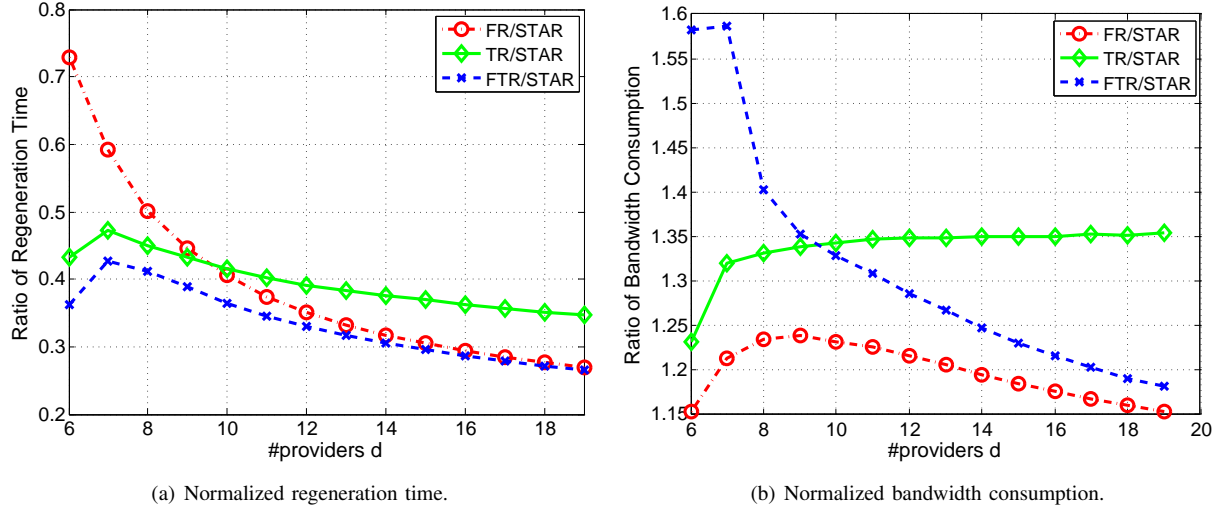


Fig. 6. Effects of d on the performances of the FR, TR and FTR schemes, in comparison with the STAR scheme based on uniform repair traffic, for $n = 20$, $k = 5$, $M = 1\text{GB}$, and uniform capacity distribution range $[10\text{Mbps}, 120\text{Mbps}]$.

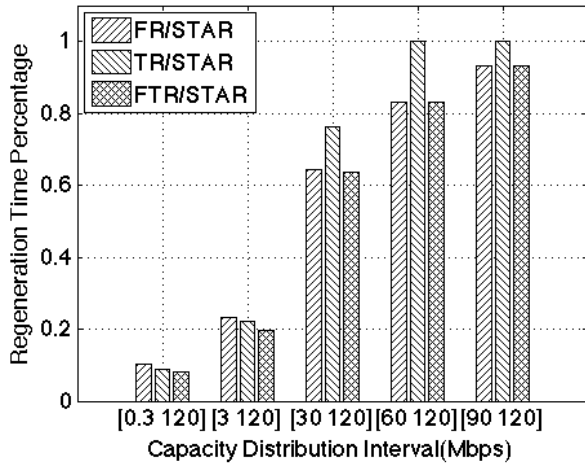
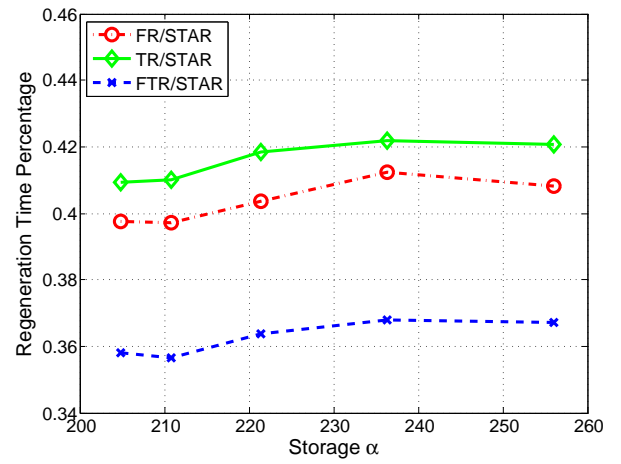


Fig. 7. Effects of network bandwidth on the regeneration time for the FR, TR, and FTR schemes, where $n = 20$, $k = 5$, $d = 10$, and $M = 1\text{GB}$.

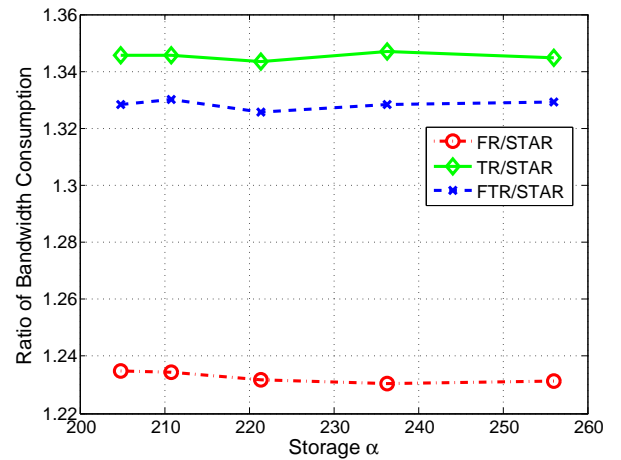
scheme does not change much as α varies. This implies that our previous conclusions for the MSR case also apply to any non-MSR case with a different storage amount α .

VII. RELATED WORK

Li *et al.* [11] first considered the heterogeneity of network bandwidth in data regeneration process and proposed a tree-structured regeneration scheme to reduce the regeneration time. They also proposed a scheme of building parallel regeneration trees to further reduce the regeneration time in the network with asymmetric links [12]. However, they only discussed the case that the regeneration scheme requires k providers, which means the minimal regeneration traffic is equal to the size of original file M . To further reduce the regeneration time, they considered the regenerating codes in the tree-structured regeneration scheme and proposed RCTREE in [7]. They employ a minimum-storage regenerating (MSR)



(a) Normalized regeneration time.



(b) Normalized bandwidth consumption.

Fig. 8. Effects of storage amount on the regeneration time for the FR, TR and FTR schemes, where $n = 20$, $k = 5$, $d = 10$, $M = 1\text{GB}$, and α varies from the MSR point to the MBR point.

codes in RCTREE, which means that the minimal regeneration traffic is $\frac{d}{k(d-k+1)}M$ bytes. Therefore, for a regeneration with d providers, each provider sends $\frac{\alpha}{d-k+1}$ blocks to its parent node. To make sure that the newcomer has enough information to restore α blocks, it has to receive data directly from at least $d-k+1$ providers. The details of how to construct an optimal regeneration tree can be found in Algorithm 1 of [7]. Although their algorithm ensures that the degree of newcomer is at least $d-k+1$, the MDS property still cannot be preserved after data regeneration.

Sun *et al.* [13] considered the scenario of repairing multiple data losses, and proposed two algorithms based on tree-structured regeneration to reduce the regeneration time. However, they assumed the same amount of data transferred between providers and newcomer for regenerating codes. According to our analysis, their regeneration schemes also cannot preserve the MDS property.

Some researches, such as [14], [15], considered the heterogeneity of nodes availability and optimized the erasure code deployment to reduce the data redundancy. Moreover, other researches, such as [16], [17], [18], [19], [6], jointly considered the repair-cost and heterogeneity of communication(download) cost on each links. They flexibly determine the amount of data to minimize the total repair cost, which is different from the regeneration time.

Regenerating codes suppose that all storage nodes store the same amount of data and the newcomer obtains the same amount of data from each provider. However, the communication cost of each provider may be different. Akhlaghi *et al.* [16] proposed a cost-bandwidth trade-off by introducing two classes of storage nodes with two different communication costs. However, the newcomer may only contact a determined number of providers in each of the two classes, and the amount of data downloaded from providers in the same class remains unchanged. Gerami *et al.* [17] considered the impact of the network topology and proposed the optimal-cost regenerating codes with variable link costs of providers with a given network topology. They assumed that, just like for conventional regenerating codes, newcomer downloads the same amount of data blocks from each provider.

The generalized repair method with various amount of information downloaded from each provider was studied by Soroush Akhlaghi *et al.* [18], Craig Armstrong *et al.* [19] and Nihar B. Shah *et al.* [6]. Armstrong *et al.* [19] proposed necessary conditions for the minimum repair bandwidth of the first two repairs at the MSR point. They conjectured that their result holds for any number of repairs, which has been proved in this paper. They also generalized their work to heterogenous storage capacities. Nihar B. Shah *et al.* [6] proposed a flexible class of regenerating codes in support of both flexible reconstruction and flexible regeneration. They accomplished flexible regeneration with two parameters γ and β_{\max} , such that the repair bandwidth of each provider can be flexibly chosen from $[0, \beta_{\max}]$ as long as the total repair bandwidth is no less than γ . Their method is generalized in this paper by introducing the concept of feasible region, which characterizes the set of feasible repair bandwidth vectors. We have also compared their work with ours in the evaluation

section above.

VIII. CONCLUSION

We have reconsidered the problem of how to reduce the regeneration time in networks with heterogeneous link capacities. We have analyzed the minimum amount of data to be transmitted on each link of the regeneration tree, and proved that the problem of building optimal regeneration tree is NP-complete. Using a proposed heuristic algorithm to construct a near-optimal regeneration tree, the regeneration time can further be reduced by allowing non-uniform end-to-end repair traffic. With the non-uniform end-to-end repair traffic, we can flexibly determine the amount of coded data generated by each provider. Simulation results have shown that our regeneration schemes are able to maintain the MDS property and reduce the regeneration time by 10% ~ 90%, compared with traditional star-structured regenerating codes. The proposed Flexible Tree-structured Regeneration scheme performs even better than RCTREE.

APPENDIX A

LOSS OF THE MDS PROPERTY IN RCTREE

Let us employ an example to demonstrate that the RCTREE scheme is unable to maintain the MDS property. Consider the overlay network shown in Fig. 1(a). Assume that a file of size $M = 480\text{Mb}$ is distributed using a $(n = 5, k = 2)$ MDS-code to the five storage nodes v_1, v_2, \dots, v_5 , of which each holds $\alpha = M/k = 240\text{Mb}$ data. The MDS property requires that the file can be reconstructed by any two storage nodes. Suppose that v_5 fails, v_0 is selected as the newcomer, and v_1, \dots, v_4 are the $d = 4$ providers. In this example, RCTREE will use the same regeneration tree as shown in Fig. 1(d), where a fixed amount of $\beta = \frac{M}{k(d-k+1)} = 80\text{Mb}$ is transmitted on each link for regeneration at v_0 .

Assume that the data collector connects to v_0 and v_3 to reconstruct the file. Fig. 9 shows the information flow graph. As marked in the figure, there is a cut of volume $2\beta + \alpha = 400\text{Mb}$, which is smaller than the file size. Therefore, the file cannot be reconstructed with storage nodes v_3, v_0 .

To find out how frequently the file reconstruction fails, we have implemented the RCTREE scheme based on Random Linear Regenerating Codes (RLRC) and run simulations with practical parameter values. The finite field $GF(2^{16})$ has been chosen for RLRC since it is sufficiently large such that the probability that linearly dependent blocks are regenerated is negligible [20]. Fig. 10 presents the simulation results for four sets of code parameter settings, showing the probability of successful file reconstruction as a function of the number of repair rounds. From this figure, we can see that the original file can hardly be reconstructed after 5 repair rounds as the number of original storage nodes becomes in turn smaller than k .

From these results, we may state that the problem of optimizing regeneration time with heterogeneous link capacities should be solved with a satisfaction to the MDS property.

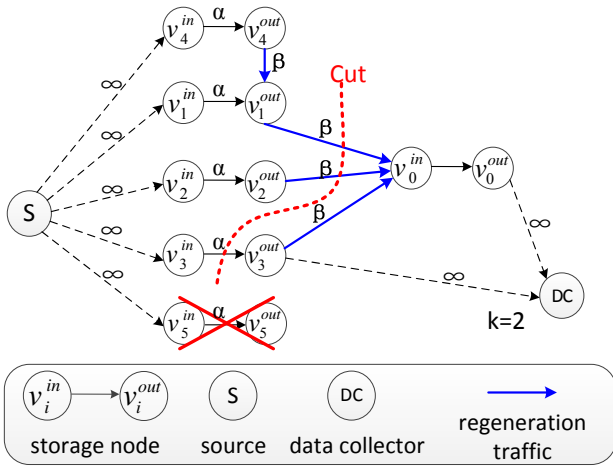


Fig. 9. An example of RCTREE and the corresponding information flow graph. The parameters are $n = 5, d = 4, k = 2, M = 480\text{Mb}, \alpha = M/k = 240\text{Mb}, \beta = \frac{M}{k(d-k+1)} = 80\text{Mb}$. A min-cut with capacity $2\beta + \alpha = 400\text{Mb}$ is illustrated by the red dashed line, implying that a DC can not reconstruct the original file by connecting with $\{v_3, v_0\}$.

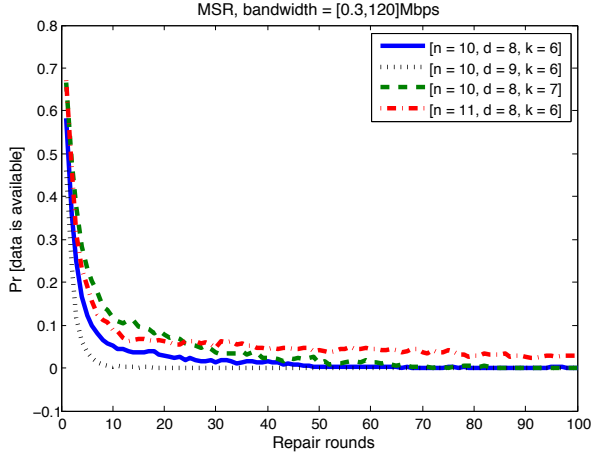


Fig. 10. Impacts of the number of repair rounds on the probability of successfully reconstructing the original file.

APPENDIX B

THE MAXIMAL FEASIBLE REGION FOR THE NON-MSR CASE

Theorem 6: For the non-MSR case of $\alpha > M/k$ and $k \geq 3$, there does not exist a maximum feasible region \mathcal{D} .

Proof: Recall that $\sigma_j(\beta)$ is defined as the sum of the $d - k + j$ smallest components of the repair bandwidth $\beta = (\beta_1, \beta_2, \dots, \beta_d)$. For a feasible region \mathcal{D} , it always holds that $\min_{\beta \in \mathcal{D}} \sigma_{j+1}(\beta) \geq \min_{\beta \in \mathcal{D}} \sigma_j(\beta)$ for $j = 1, \dots, d-1$. Let i denote the number of terms $\min_{\beta \in \mathcal{D}} \sigma_j(\beta) \geq \alpha$, i.e.,

$$\min_{\beta \in \mathcal{D}} \sigma_{k-i}(\beta) < \alpha \quad \wedge \quad \min_{\beta \in \mathcal{D}} \sigma_{k-i+1}(\beta) \geq \alpha,$$

where i must range from 1 to k , since $\sigma_k(\beta)$ must be no less than α for a successful repair. Therefore, all feasible regions can be partitioned into k groups by the value of i , and for $\alpha > M/k$, every group is non-empty. In order to prove this

theorem, it is sufficient to show that in the i -th group, where $1 \leq i \leq k-2$ and $M - i\alpha > 0$, there does not exist a maximum region. Note that we call a feasible region *maximum*, if it includes all feasible regions. A feasible region \mathcal{D} is *maximal*, if adding any vector $\beta \in \mathbb{R}^d \setminus \mathcal{D}$ to \mathcal{D} makes it infeasible.

We prove this by contradiction. If there exists a maximum region \mathcal{D}_{max} in the i -th group, it must contain all the vectors β satisfying the following constraints:

$$\begin{aligned} \sum_{j=1}^{k-i} \sigma_j(\beta) &\geq M - i\alpha \\ \sigma_{k-i}(\beta) &< \alpha \\ \sigma_{k-i+1}(\beta) &\geq \alpha \\ \forall j : \beta_j &\leq \alpha. \end{aligned}$$

Then it is sufficient to prove that \mathcal{D}_{max} does not satisfy the min-cut condition.

Pick up a $\beta \in \mathcal{D}_{max}$ and assume that $\beta_1 \leq \beta_2 \leq \dots \leq \beta_d$ without loss of generality. Under this assumption, $\sigma_j(\beta) = \beta_1 + \dots + \beta_{d-k+j}$. If $\sum_{j=1}^{k-i} \sigma_j(\beta) > M - i\alpha$, we construct β^0 as follows:

$$\beta_j^0 = \begin{cases} t\beta_j & \text{if } 1 \leq j \leq d-i \\ \alpha & \text{if } d-i+1 \leq j \leq d \end{cases}$$

where $t = \frac{M-i\alpha}{\sum_{j=1}^{k-i} \sigma_j(\beta)}$. Therefore,

$$\begin{aligned} \sum_{j=1}^{k-i} \sigma_j(\beta^0) &= t \sum_{j=1}^{k-i} \sigma_j(\beta) = M - i\alpha \\ \sigma_{k-i}(\beta^0) &< \sigma_{k-i}(\beta) < \alpha \\ \sigma_{k-i+1}(\beta^0) &\geq \alpha \\ \forall j : \beta_j^0 &\leq \alpha. \end{aligned}$$

Thus, $\beta^0 \in \mathcal{D}_{max}$. Let m be the minimum integer such that $\sigma_m(\beta^0) > 0$. As $M - i\alpha > 0$, we have $m \leq k - i$. Thus, it is in turn sufficient to prove

$$\sum_{j=1}^{k-i} \min_{\beta \in \mathcal{D}_{max}} \sigma_j(\beta) < \sum_{j=1}^{k-i} \sigma_j(\beta^0) = M - i\alpha.$$

Because $\min_{\beta \in \mathcal{D}_{max}} \sigma_j(\beta) \leq \sigma_j(\beta^0)$, we will show that $\min_{\beta \in \mathcal{D}_{max}} \sigma_m(\beta) < \sigma_m(\beta^0)$ to complete the proof. Due to the definition of m , we have $\beta_1 = \beta_2 = \dots = \beta_{d-k+m-1} = 0$, and $\beta_{d-k+m} > 0$. Then we construct β' as follows:

$$\beta'_j = \begin{cases} \beta_j^0 - \epsilon & \text{if } j = d - k + m \\ \beta_j^0 + (k-i)\epsilon & \text{if } j = d - k + k - i \\ \beta_j^0 & \text{otherwise} \end{cases}$$

where $0 < \epsilon < \min\{\frac{\alpha - \sigma_{k-i}(\beta^0)}{k-i-1}, \beta_{d-k+m}^0\}$.

Therefore,

$$\begin{aligned}
\sum_{j=1}^{k-i} \sigma_j(\beta') &= \sum_{j=1}^{m-1} \sigma_j(\beta^0) + \sum_{j=m}^{k-i-1} (\sigma_j(\beta^0) - \epsilon) \\
&\quad + \sigma_{k-i}(\beta^0) + (k-i-1)\epsilon \\
&\geq \sum_{j=1}^{k-i} \sigma_j(\beta^0) = M - i\alpha \\
\sigma_{k-i}(\beta') &= \sigma_{k-i}(\beta^0) + (k-i-1)\epsilon < \alpha \\
\sigma_{k-i+1}(\beta') &= \sigma_{k-i+1}(\beta^0) + (k-i-1)\epsilon \geq \alpha \\
\forall j : \beta'_j &\leq \alpha,
\end{aligned}$$

which means $\beta' \in \mathcal{D}_{max}$ and $\min_{\beta \in \mathcal{D}_{max}} \sigma_m(\beta) \leq \sigma_m(\beta') = \sigma_m(\beta^0) - \epsilon$. ■

REFERENCES

- [1] P. P. C. L. Yuchong Hu, Henry C. H. Chen and Y. Tang, "Nccloud: applying network coding for the storage repair in a cloud-of-clouds," in *Proceedings of USSENIX Conference on File and Storage Technologies (FAST)*, 2012.
- [2] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5. ACM, 2003, pp. 29–43.
- [3] A. G. Dimakis, P. B. Godfrey, M. J. W. Y. Wu, and K. Ramchandran, "Network Coding for Distributed Storage System," *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.
- [4] B. Gastón, J. Pujol, and M. Villanueva, "A realistic distributed storage system: the rack model," *arXiv preprint arXiv:1302.5657*, 2013.
- [5] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, ser. IMC '10. ACM, 2010, pp. 267–280.
- [6] N. Shah, K. V. Rashmi, and P. Kumar, "A flexible class of regenerating codes for distributed storage," in *Proceedings of IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2010.
- [7] J. Li, S. Yang, X. Wang, and B. Li, "Tree-structured Data Regeneration in Distributed Storage Systems with Regenerating Codes," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2010.
- [8] S.-J. Lee, P. Sharma, S. Banerjee, S. Basu, and R. Fonseca, "Measuring bandwidth between planetlab nodes," in *Passive and Active Network Measurement*. Springer, 2005, pp. 292–305.
- [9] I. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.
- [10] R. C. Prim, "Shortest connection networks and some generalizations," *Bell system technical journal*, vol. 36, no. 6, pp. 1389–1401, 1957.
- [11] J. Li, S. Yang, X. Wang, X. Xue, and B. Li, "Tree-structured data regeneration with network coding in distributed storage systems," in *Proceedings of 17th International Workshop on Quality of Service (IWQoS)*, 2009.
- [12] J. Li, S. Yang, and X. Wang, "Building parallel regeneration trees in distributed storage systems with asymmetric links," in *2010 6th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2010.
- [13] W. Sun, Y. Wang, and X. Pei, "Tree-structured parallel regeneration for multiple data losses in distributed storage systems based on erasure codes," *Communications, China*, vol. 10, no. 4, pp. 113–125, 2013.
- [14] L. Pamies-Juarez, P. Garcia-Lopez, and M. Sanchez-Artigas, "Heterogeneity-aware erasure codes for peer-to-peer storage systems," in *International Conference on Parallel Processing (ICPP)*, 2009.
- [15] G. Xu, S. Lin, G. Wang, X. Liu, K. Shi, and H. Zhang, "Hero: Heterogeneity-aware erasure coded redundancy optimal allocation for reliable storage in distributed networks," in *International Performance Computing and Communications Conference (IPCCC)*, 2012.
- [16] S. Akhlaghi, A. Kiani, and M. R. Ghanavati, "Cost-bandwidth Tradeoff in Distributed Storage Systems," *Computer Communications*, vol. 33, no. 17, pp. 2105–2115, 2010.
- [17] M. Gerami, M. Xiao, and M. Skoglund, "Optimal-cost Repair in Multi-hop Distributed Storage Systems," in *Proc. of IEEE International Symposium on Information Theory (ISIT)*, 2011, pp. 1437–1441.
- [18] S. Akhlaghi, A. Kiani, and M. Ghanavati, "A fundamental trade-off between the download cost and repair bandwidth in distributed storage systems," in *Proceedings of IEEE International Symposium on Network Coding (NetCod)*, 2010.
- [19] C. Armstrong and A. Vardy, "Distributed storage with communication costs," in *Proceedings of Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2011.
- [20] A. Duminuco and E. Biersack, "A practical study of regenerating codes for peer-to-peer backup systems," in *29th IEEE International Conference on Distributed Computing Systems (ICDCS'09)*. IEEE, 2009.